CMMI for Embedded Systems Development

O.Univ.-Prof. Dipl.-Ing. Dr. Wolfgang Pree
Software Engineering Gruppe

Leiter des Fachbereichs Informatik
cs.uni-salzburg.at

UNIVERSITÄT SALZBURG

Inhalt

- Projekt-Kontext
  - CMMI
  - FIT-IT-Projekt mit AVL-List
- erwartete Ergebnisse und Zeitrahmen
  - Prozessbeschreibungen für Stufen 2 – 3
  - Trainingsunterlagen

Was ist CMMI?

Capability Maturity Model® Integration (CMMI)

- **weltweiter Standard** zur Beurteilung des Reifegrades der Software-Entwicklungsprozesse
  - vom Software Engineering Institute der Carnegie Mellon University definiert http://www.sei.cmu.edu/cmmi/
- Software-/System-Lieferanten müssen verstärkt **Qualitätsvorgaben der Auftrageber** einhalten (zB CMMI-Reifegrad 2 bis 2007).
- Um Embedded Software-Entwicklung CMMI-konform zu machen, sind deren Spezifika zu berücksichtigen.

CMMI in a nutshell (laut SEI)

A capability maturity model **delineates the characteristics of a mature, capable process.**

- It **identifies the practices** that are basic to implementing effective processes as well as advanced practices.
- It also **assigns to those practices associated maturity levels** ranging from unrepeatable to a mature, well-managed process.
- Typically **a path is recommended through the various practices for achieving higher levels of maturity** and, therefore, improve an organization's processes.

FIT-IT-Projekt embeddedCMMI

embeddedCMMI-Projekt

FIT-IT Embedded Systems

- unterstützt durch FIT-IT Embedded Systems (www.fit-it.at), einer BMvit-Initiative

- Rahmen: ca. 120 k€
- Laufzeit: April 2005 – Juli 2006
- Industriepartner: AVL List GmbH, Graz
- ein Großteil der Ergebnisse wird öffentlich zugänglich sein

Wirtschaftlicher Nutzen für die Industrie

Die aufwändige Analyse und CMMI-konforme Beschreibung von Prozessen wird allen interessierten Unternehmen zur Verfügung gestellt

- fundierte Beschreibung von CMMI-Prozessen, die typisch für Embedded-Software-Entwicklung sind
- diese sind von Unternehmen, die Embedded Software entwickeln, sofort oder mit minimalen Anpassungen umsetzbar
- Trainingsunterlagen erleichtern den “Kulturwechsel” innerhalb eines Unternehmens

Ein Unternehmen sollte nahe Stufe 2 sein, um optimal von den Ergebnissen zu profitieren

- **CoPro-Ansatz** (Wallmüller/Pree/Templ): Code UND Prozesse betrachten
  - **Prozesse**: CMMI-Assessment
  - **Code**: Software-Qualitätsanalyse (automatisch und manuell)
- Aufwand vor Ort bei einem Unternehmen: ca. 4–8 Tage

Wissenschaftliche Herausforderungen bei der CMMI-Anpassung

Bidirektionale Verfolgbarkeit von Anforderungen

- Anforderungen an Embedded Software (zB “feels like a Porsche”) sind schwer zu erfassen. CMMI fordert, dass diese von der Spezifikation über den Entwurf bis hin zur Implementierung bidirektional verfolgt werden können.

  Eine CMMI-konforme UND praktikable Handhabung zu definieren, erfordert innovative Ansätze. Ein wissenschaftlich fundiertes Software Engineering Know-How ist dazu Voraussetzung.

Einfluß von Code-Generatoren (Simulink)

- Aus Simulink-Modellen wird C-Code generiert. Das hat profunde Auswirkungen auf den Entwicklungsprozess.

  Diese Auswirkungen müssen analysiert und adäquat beschrieben werden, um CMMI-konform vorzugehen. Voraussetzung dazu ist konzeptionelles Verständnis sowohl der CMMI-Prozessvorgaben als auch der Simulink-Werkzeuge.

Stark variierende Entwicklungszyklen

- Je nachdem ob und wie die Embedded Software simuliert wird (in Simulink; mit einem Hardware-in-the-Loop-Simulator) oder im Gerät integriert wird, variiert die Länge und der Detailierungsgrad des Entwicklungszyklus.

  Das ist bei einer CMMI-konformen Beschreibung der Prozesse entsprechend zu berücksichtigen. Voraussetzung dazu ist fachliches Verständnis der Auswirkungen der verschiedenen Laufzeitumgebungen.

embeddedCMMI project plan

Expected project results

- Detailed CMMI-compatible process descriptions up to maturity level 3. These serve as templates for companies which have to adhere to CMMI, for example, because the car industry demands that.
- Description of the characteristics of embedded software development that imply adapted CMMI processes.
- Evaluation of tools that support the adapted CMMI processes.
- Metrics for measuring the effects of the process improvements.
- Education and training program.

covered CMMI process areas

- PA Requirements Management (REQM/ML2)
- PA Project Planning (PP/ML2)
- PA Project Monitoring & Control (PMC/ML2)
- PA Supplier Agreement Management (SAM/ML2)
- PA Measurement and Analysis (MA/ML2)
- PA Process and Product Quality Assurance (PPQA/ML2)
- PA Configuration Management (CM/ML2)
- PA Requirements Development (RD/ML3)
- PA Technical Solution (TS/ML3)
- PA Product Integration (PI/ML3)
- PA Verification (VER/ML3)
- PA Validation (VAL/ML3)
- PA Organizational Process Focus (OPF/ML3)
- PA Organizational Process Definition (OPD/ML3)
- PA Organizational Training (OT/ML3)
- PA Integrated Project Management (IPM/ML3)
- PA Risk Management (RSKM/ML3)
- PA Decision Analysis and Resolution (DAR/ML3)

Project activities

- Activity 1:

  duration: 4/05 – 12/05

  - **identification of CMMI process areas that need to be adapted** to be best fitted for development of embedded SW (work is based on CMMI process descriptions developed by AVL and provided as inputs into this project)
  - planning and accomplishment of CMMI process improvements for the development of embedded SW

- Activity 2:

  duration: 7/05 – 2/06

  - **evaluation of tools** that support the adapted CMMI processes
  - **investigation of metrics** for measuring the improvements (input: CMMI metrices defined by AVL)

Project activities

- Activity 3:

  duration: 8/05 – 7/06

  - sample setup of organizational environment for integration

- Activity 4:

  duration: 10/05 – 7/06

  - **preparation of education and training material** for CMMI based development in embedded SW projects
  - Perform training for CMMI based development of embedded SW using the training and educational materials developed above

Project activities

- Activity 5:

  duration: 11/05 – 7/06

  - **description of Simulink patterns** to develop SW for embedded systems using SW code generators.

- Activity 6:

  duration: 11/05 – 7/06

  - development of **methods to ensure bi-directional traceability of requirements in embedded SW development**.